An Automated Solution to the Multiuser Carved Data Ascription Problem

Simson L. Garfinkel, Aleatha Parker-Wood, Daniel Huynh, and James Migletz

***Abstract*—This paper presents a novel solution to the problem of determining the ownership of carved information found on disk drives and other storage media that have been used by more than one person. When a computer is subject to forensic examination, information may be found that cannot be readily ascribed to a specific user. Such information is typically not located in a specific file or directory, but is found through file carving, which recovers data from unallocated disk sectors. Because the data is carved, it does not have associated file system metadata, and its owner cannot be readily ascertained. The technique presented in this paper starts by automatically recovering both file system metadata as well as extended metadata embedded in files (for instance, embedded timestamps) directly from a disk image. This metadata is then used to find exemplars and to create a machine learning classifier that can be used to ascertain the likely owner of the carved data. The resulting classifier is well suited for use in a legal setting since the accuracy can be easily verified using cross-validation. Our technique also results in a classifier that is easily validated by manual inspection. We report results of the technique applied to both specific hard drive data created in our laboratory and multiuser drives that we acquired on the secondary market. We also present a tool set that automatically creates the classifier and performs validation.**

***Index Terms*—Data mining, forensics, information security.**

I. Introduction

MANY computers are routinely used by more than one person. Families, roommates, and coworkers frequently share a single computer. In many schools there are “classroom computers” that are shared by each teacher that is assigned to teach in the classroom.

The idea for this research project came from an investigation of child pornography discovered on a computer that was shared by two different teachers. The pornography was not found in files or directories belonging to one user or the other: instead, it was found in a deleted file for which the original name and owner could not be determined. This commonly happens when a file is deleted and the original directory entry is overwritten, forcing the file to be recovered through *file carving* [10], [15].

The computer forensics investigator assigned to the case was given the task of trying to establish whether the pornography had been downloaded by user A or user B. We call this process the *owner ascription problem*. Before we completed this research, there was no principled procedure for performing this ascription. Instead, the forensics examiner solved the problem by trying to find specific *features* of the file that were in common with other files on the computer that could be ascribed to user A or user B. Once a few common features were found, a match was declared. Unfortunately, this process is error-prone, because it considers neither all of the available features nor all of the available files to which the file in question can be matched.

The problem of ascription is neither new nor unique to digital security and forensics. Since the 18th century, scholars have attempted to alternately assign or repudiate authorship of Shakespeare’s plays and poems. Unlike text attribution, however, forensic investigators are frequently asked to ascribe not *authorship*, but *agency*—that is, the police investigator in the pornography case needed to infer which of the computer’s two authorized users had placed the digital photographs on the computer’s hard drive, not the identity of the photographer.

Based on our interactions with the forensic investigator, we developed an automated approach for ascribing carved data to a specific user of a multiuser computer system. Our approach is superior to the manual approach because it considered *all* of the available data on the computer’s hard drive, and because it gives an error rate that is specific to the hard drive in question. Thus, unlike the investigator’s procedure, our approach is compliant with the U.S. Supreme Court’s *Daubert* ruling, which holds that scientific methods used in court must have a “known or potential rate of error,” [18] and must be subjected to empirical testing, which we perform in this paper.

Our approach combines automated feature extraction combined with straightforward application of unsupervised machine learning. The approach specifically does not rely upon the examiner to chose which features to use, because few examiners would have the necessary knowledge to do so, and the most effective features may vary from drive to drive. Instead, the approach uses *all* of the available features that our tools can extract, and determines the accuracy of the resultant classifier using leave-one-out validation.

A. Hypothesis and Contributions

We hypothesize that there are characteristic similarities among all information deposited on a storage device used by a computer user, and that these characteristics will have different

Manuscript received May 17, 2010; accepted July 09, 2010. Date of publication July 23, 2010; date of current version November 17, 2010. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Nasir Memon.

S. L. Garfinkel is with the Department of Computer Science, Naval Postgraduate School, Pacific Grove, CA 93950 USA (e-mail: simsong@acm.org).

A. Parker-Wood is with the Department of Computer Science, University of California, Santa Cruz, CA 95060 USA.

D. Huynh is with the Department of Computer Science, United States Military Academy, West Point, West Point, NY 10096 USA.

J. Migletz is with the U.S. Marine Corps, Quantico, VA 22134 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2010.2060484

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **MAY 2010**	2. REPORT TYPE	3. DATES COVERED **00-00-2010 to 00-00-2010**

4. TITLE AND SUBTITLE **An Automated Solution to the Multiuser Carved Data Ascription Problem**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Postgraduate School,Department of Computer Science,Monterey,CA,93943**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This paper presents a novel solution to the problem of determining the ownership of carved information found on disk drives and other storage media that have been used by more than one person. When a computer is subject to forensic examination information may be found that cannot be readily ascribed to a specific user. Such information is typically not located in a specific file or directory, but is found through file carving, which recovers data from unallocated disk sectors. Because the data is carved, it does not have associated file system metadata, and its owner cannot be readily ascertained. The technique presented in this paper starts by automatically recovering both file system metadata as well as extended metadata embedded in files (for instance, embedded timestamps) directly from a disk image. This metadata is then used to find exemplars and to create a machine learning classifier that can be used to ascertain the likely owner of the carved data. The resulting classifier is well suited for use in a legal setting since the accuracy can be easily verified using cross-validation. Our technique also results in a classifier that is easily validated by manual inspection. We report results of the technique applied to both specific hard drive data created in our laboratory and multiuser drives that we acquired on the secondary market. We also present a tool set that automatically creates the classifier and performs validation.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**15**	

(and unique) values for different computer users—even when the users are using the same computer. We further hypothesize that these characteristics can be identified and used to automatically ascribe residual information to the user responsible for the information's presence.

In this paper, we demonstrate the approach using both *realistic* data that was created in the laboratory, as well as *real* data that was obtained from secondary computers purchased on the secondary market. For each drive, we construct a classifier using all of the available data on the disk except for one file; we then use the classifier to classify the file that has been left out. We then repeat this process for every file on the disk. This technique, called *leave-one-out validation*, produces a confusion table that shows how the classifiers classified the files known to belong to each user. We assume that carved data would classify with similar accuracy rates, since the only difference between carved data and resident files is that carved data was once a file that was subsequently deleted. (On systems that have high levels of system activity or disks that are nearly filled, it is likely that carved data will have been recently deleted. But on systems that have low levels of system activity or disks that are mostly empty, carved data can be quite old—it may even predate the disk's file system.)

By making a modification to the approach and generating additional confusion tables after each modification, we are able to increase the overall accuracy of the approach. Additional testing shows that these improvements are reflected in many drives that are tested.

Because our approach tests the hypothesis *on every hard drive to which it is applied*, it produces different accuracy and error rates for different hard drives. This is as it should be: some hard drives have data that cluster well, while other hard drives have data that do not.

Our approach does not say whether or not a specific file was created by a specific user—it only reports how well the file clusters with other files created by that user. In drives with high accuracy rates, the results could be reported in court for the trier of fact to consider. In cases where the leave-one-out validation shows low accuracy, the results are inconclusive and should not be reported in court or used as the basis of a future investigation.

For obvious reasons, we do not report the results of applying the approach to actual case data.

B. Importance and Applicability

The contribution in this paper is both important to the matter at hand and applicable to other situations. Although it is certainly common for business professionals to have a laptop and/or a desktop computer for their exclusive use, even today it is relatively common for a single computer to be used by multiple individuals. This is the case in many schools, where a single classroom or laboratory computer might have several authorized users. It is also common in many families, as well as in libraries, hospitals, and Internet cafes.

Another way for a computer to be used by more than one person is for the computer to be stolen and later recovered. Finally, although the automated approach described in this paper is only applied to a single computer, we believe that it could be applied to multiple computers or to ascribing information found on cloud-based servers or portable storage devices. We will discuss this possibility more in Section VII-D4.

C. Related Work

Very little work has been done in the area of automated file ascription for multiuser hard drives, but there has been a lot of research into ascription in the broader sense, which we discuss briefly here. There are also manual techniques which law enforcement currently uses to perform ascription, although we have been unable to find any publicly available descriptions of these procedures.

Cross-Drive Analysis: The initial work on automated forensic feature extraction was part of Garfinkel's Cross-Drive Analysis and Forensic Feature Extraction [5]. Garfinkel's work was primarily concerned with the identification of previously unknown social networks or links between individuals and organizations, rather than the ascription of content discovered on drives.

Text Mining for Ascription: One common method of ascribing documents to a specific person is to apply statistical linguistic techniques to the content [9]. This technique has been used with a great deal of success for ascribing historical documents (for instance, the authors of the disputed Federalist Papers [16], [1]). But this technique is only effective for text, not for general computer usage and files. It is only for ascribing the creator of content, rather than the owner.

Lazy Decision Trees: The approach of creating a decision tree based on the data being tested was introduced as Lazy Decision Trees (LazyDT) by Friedman, Kohavi, and Yun [4]. As with our use of C4.5, LazyDT builds a decision tree for each instance to be classified, ignoring dimensions of the data set that are missing in the instance. In their paper, LazyDT is shown to outperform C4.5 by approximately 2%, largely because of its improved handling of missing values. Given that we are also handling missing values, the main advantage of adopting LazyDT would most likely be improved performance.

Accountability Systems: Although there exist practical systems for achieving accountability in distributed systems such as PeerReview [8], these systems require the use of specialized software or runtime environments. Computer crime investigators do not have the luxury of deciding the software that will be used by offenders. Although systems may exist that eliminate the multiuser carved data ascription problem through the use of append-only logging file systems, offenders are more likely to be using computers running Microsoft Windows or MacOS.

Willassen's Ph.D. thesis [19] discusses the use of native timestamps on modern computer systems to improve opportunities for evidence reconstruction and user accountability. Willassen discusses the fundamental problem with using timestamps—that they are sometimes subject to manipulation by the user—and proposes approaches for resolving that problem through the use of a hypothesis-based approach. In his thesis, Williassen only applies his approaches to systems that are used by a single user. If they were extended to multiuser systems, those approaches could be used in conjunction with the technique that we introduce in the remainder of the paper.

D. Outline of Paper

This paper introduces an important problem that confronts computer forensic investigators, but which has been largely ignored by developers of forensic tools: how to automatically ascribe information found on a subject computer to a specific user when that computer has been used by multiple individuals (Section II). We propose a novel solution based on automated feature extraction and machine learning, and evaluate the solution on two "realistic" hard drive disk images consisting of data generated by experimenters playing the role of multiple users (Section III). The approach is refined through the use of dimensional reduction—specifically by projecting all of the timestamps associated with user activity present in the metadata to a single timeline (Section V). We then validate the technique using a set of real disks with real user data that were purchased on the secondary market (Section VI).

Section VIII presents our conclusions, including applications to current law enforcement tasks and the opportunities for future work.

II. Multiuser Carved Data Ascription Problem

Carved data can be tremendously important in forensic investigations. But when a computer is used by more than one person, carving can present the examiner with a difficult problem. Data that is recovered through carving does not have an associated file name, file system timestamps, ownership, file permissions, or other file system metadata that might be used to connect the data with one of the computer's users. Carved data must, as a result, be manually attributed, or *ascribed*, to a specific individual.

Consider the case of a classroom computer used by teacher1 during lunch on Mondays and Tuesdays and by teacher2 during lunch on Wednesdays and Thursdays. If the computer is subject to a forensic examination on Friday and child pornography is found through file carving, there is no readily apparent way to ascribe the pornography to either teacher1 or teacher2.

We call this problem the *multiuser carved data ascription problem* (hereafter shortened to the "Ascription Problem"). In the area of law enforcement, the Ascription Problem arises when a single computer is used by multiple individuals. This can be the case when two people share one computer, or when a computer is stolen, used by a second person, and later subjected to a forensic inspection. When the defendant claims "The other person put the data there," the investigator's only recourse is to find some kind of information that ties the carved data to the suspect. This additional information can be *file system metadata, file placement information*, and *embedded file metadata*:

File system metadata is information stored by the file system externally to the file—for example, the modification time of files. By assembling all of the timestamps that are ascribable to each individual that has used the storage device, it is possible to create a timeline that indicates when each person used the device.

File placement information is the physical sector numbers where data is stored. This is useful in situations where a region of the device was used to preferentially store files belonging to one user.

Embedded file metadata is information found inside the carved data itself. For example, many digital cameras store the camera's serial number inside each image that the camera takes. Finding a serial number in JPEGs that are clearly ascribable to one user and finding the same serial number in a carved JPEG would link the carved JPEG to the user.

Finding, tabulating, and evaluating all of the available metadata, both embedded and from the file system, is a daunting task with the amount of data and diversity of data types on a typical hard drive (see Table I). This task can be complicated by the fact that there may be conflicting information on the storage device—some metadata that implicates one user, and some that implicates another. With manual correlation an analyst can try to evaluate all of the available metadata and claim that one person is guilty and another is innocent, but doing so is a mind-numbing and error-prone exercise. Manual examination also makes the examination extraordinarily invasive.

III. Basic Approach

Our approach automates the procedure outlined above, using specialized metadata extractors and automated classification techniques to process the drive:

1) Forensic software analyzes the disk image and locates all of the ascribable *exemplar files*—recoverable allocated and deleted files in the file system for which the owner can be reliably ascertained.
2) For each exemplar, file system metadata and file placement information is gathered from the intact file system, and embedded metadata is gathered using format-specific *metadata extractors*. All of this information for all of the exemplars is combined into a single file.
3) The data is preprocessed.
4) The exemplars are used to train a classifier.
5) The classifer's accuracy is determined using leave-one-out validation.
6) The carved data that needs to be ascribed is processed with the metadata extraction system.
7) Finally, the carved data's metadata and placement information is processed using the classification system.

We will now discuss each step in detail.

A. Disk Image Analysis and Feature Extraction

We created a program called `fiwalk` [6] (*f*ile and *i*node *walk*er) to ingest each disk image, automatically process all partitions that might be present in the disk image, examine all of the files and inodes in each one, and produce an output file consisting of a list of each file (both allocated and deleted) and all collected metadata. During the disk image analysis, metadata is collected for each file. Collected file system metadata includes each file's name, the file create, modify, access and inode change times, the file's location in the disk image, the number of fragments.

The embedded metadata is extracted from each intact file using `fiwalk`'s metadata extraction system [14]. For this paper, we use metadata extractor plug-ins for Microsoft Office, OpenOffice, JPEG, and other multimedia file formats. An example of the variety extractable metadata attributes is shown in Fig. 1.

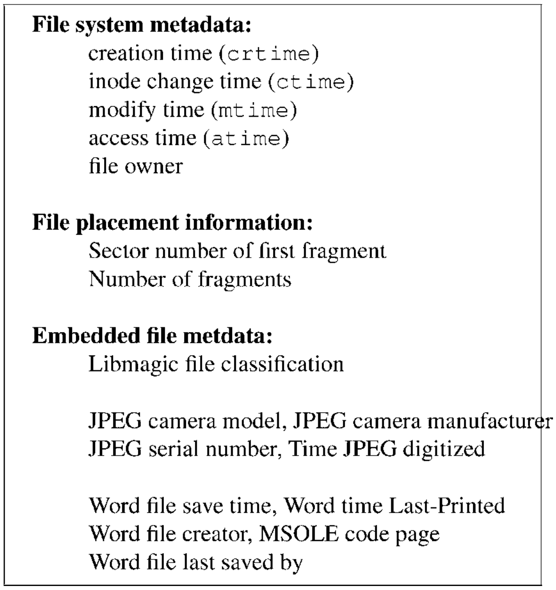
TABLE I
Actual Attributes and Their Types Extracted from the DomexUsers Disk Image Using fiwalk. Some of These Attributes Are File System Metadata, Some Are File Placement Information, but Most Are Embedded File Metadata Extracted from Digital Photographs, Digital Music, and Microsoft Office File Formats. All Attributes Are Provided to the Machine Learning Algorithm; There Is no Manual Pruning

Attribute	Type	Attribute	Type	Attribute	Type
-Clips	numeric	FlashPixVersion	string	Subject-Distance-Range	string
ALLOC	numeric	Focal-Length	string	SubsecTime	numeric
Aperture	string	Focal-Length-In-35mm-Film	numeric	Template	string
Archive-File1	string	Focal-Plane-Resolution-Unit	string	Thumbnail	string
Archive-File10	string	Focal-Plane-x-Resolution	numeric	ThumbnailSize	numeric
Archive-File11	string	Focal-Plane-y-Resolution	numeric	Title	string
Archive-File12	string	GPS-tag-version	string	UNALLOC	numeric
Archive-File13	string	Gain-Control	string	USED	numeric
Archive-File14	string	Gamma	numeric	Unknown1	numeric
Archive-File15	string	Generator	string	Unknown3	string
Archive-File16	string	ISO-Speed-Ratings	numeric	Unknown6	string
Archive-File17	string	Image-Description	string	Unknown7	numeric
Archive-File18	string	InteroperabilityIndex	string	User-Comment	string
Archive-File19	string	InteroperabilityVersion	numeric	White-Balance	string
Archive-File2	string	Keywords	string	White-Point	string
Archive-File20	string	Last-Modified	date	XP-Title	string
Archive-File21	string	Last-Printed	date	YCbCr-Coefficients	string
Archive-File22	string	Last-Saved-by	string	YCbCr-Positioning	string
Archive-File3	string	LastSavedBy	string	atime	date
Archive-File4	string	Light-Source	numeric	comment	string
Archive-File5	string	Links-Dirty	string	content-type	string
Archive-File6	string	Manager	string	crtime	date
Archive-File7	string	Manufacturer	string	ctime	date
Archive-File8	string	MaxApertureValue	string	description	string
Archive-File9	string	Metering-Mode	string	filename	string
Artist	string	Model	string	filesize	numeric
CFA-Pattern	numeric	Number-of-Bytes-in-the-Document	numeric	flags	numeric
COMPRESSED	numeric	Number-of-Characters	string	format	string
Color-Space	string	Number-of-Hidden-Slides	numeric	frag1startsector	numeric
Company	string	Number-of-Lines	string	frag2startsector	numeric
ComponentsConfiguration	string	Number-of-Notes	numeric	fragments	numeric
Compressed-Bits-per-Pixel	numeric	Number-of-Pages	string	genre	string
Compression	string	Number-of-Paragraphs	string	gid	numeric
Contrast	string	Number-of-Slides	numeric	id	numeric
Copyright	string	Number-of-Words	string	inode	numeric
Created	date	Orientation	string	libmagic	string
Creator	string	Paragraph-Revision-ID-Default1	string	md5	string
Custom-Rendered	string	Paragraph-Revision-ID1	string	mimetype	string
Date-and-Time	date	PixelXDimension	numeric	mode	numeric
Date-and-Time–digitized-	date	PixelYDimension	numeric	msole-codepage	numeric
Date-and-Time–original-	date	Primary-Chromaticities	string	mtime	date
Description	string	Resolution-Unit	string	nlink	numeric
Digital-Zoom-Ratio	numeric	Revision	string	orientation	string
Document-Pairs	string	Saturation	string	partition	numeric
Document-Parts	string	Scale	string	resolution	string
Editing-Duration	date	Scene-Capture-Type	string	seq	numeric
Exif-Version	string	Scene-Type	numeric	sha1	string
Exposure-Bias	string	Security-Level	numeric	size	string
Exposure-Mode	string	Sensing-Method	string	software	string
Exposure-Time	string	Sharpness	string	type	numeric
ExposureProgram	string	Shutter-speed	string	uid	numeric
FNumber	string	Software	string	x-Resolution	numeric
File-Source	string	SubSecTimeDigitized	numeric	y-Resolution	numeric
Filename	string	SubSecTimeOriginal	numeric		
Flash	string	Subject	string		

The embedded metadata extracted is sparse and has many missing values. For example, the typical Microsoft Word file might contain metadata consisting of the author and organization, creation time, last printed time, and so on. For a digital image, the metadata might include the shutter speed and camera serial number. Clearly, Word files will not have JPEG metadata, and vice-versa. Although attributes with many missing values can be a problem for some machine learning algorithms, our implementation needs to focus on these attributes as they are frequently correlated with the circumstances surrounding the document's creation and use.

B. *Exemplar Selection*

Exemplars are selected from the disk image based on the availablity of file system metadata denoting file ownership. Our system currently determines the owner from the name of the



File system metadata:
creation time (`crtime`)
inode change time (`ctime`)
modify time (`mtime`)
access time (`atime`)
file owner

File placement information:
Sector number of first fragment
Number of fragments

Embedded file metdata:
Libmagic file classification

JPEG camera model, JPEG camera manufacturer
JPEG serial number, Time JPEG digitized

Word file save time, Word time Last-Printed
Word file creator, MSOLE code page
Word file last saved by

Fig. 1. Examples of features used by the automated ascription system.

containing directory. For example, the system infers that the Windows file `/Documents and Settings/sally/My Documents/Presentation.ppt` is owned by the user "sally" and the Macintosh file `/Users/jason/house.jpg` is owned by "jason." Currently our system ignores Unix and Windows file permissions and access control lists, because of incomplete support for this metadata by Sleuth Kit [2], on which `fiwalk` relies.

C. Feature Selection

For this work, our algorithm employs no specific feature selection, weighting or biasing, although the C4.5 algorithm implicitly determines which are the most important features to use when it constructs a decision tree. Our goal for this research is to allow the automated ascription of carved data—for example, a system that could be run in a police station by an forensic examiner with no formal training in data mining. We would like to have the system automatically determine if, for example, the serial number of JPEGs or a custom Microsoft Word metadata field from a document management system should be considered as part of the ascription process. Because there are hundreds of extractable features on a typical hard drive, we conclude that the best approach is to simply use *all* of the features and allow the algorithm to determine which features are the most useful. Although in the future it might be possible to increase the accuracy of this technique with some kind of automated feature selection or weighting, our experience will show that such automation is not necessary to achieve reliable results.

D. Data Preprocessing

Taking into account the preceding, our data preprocessing consists of these steps:

1) The owner of each file is determined using the algorithm described above to extract the owner from the file path. This owner is added to the ARFF file as a new attribute for each file.
2) Attributes that are unique for each entry (the `fiwalk` id, filename, and file hashes) that cannot be used to link files together are discarded.
3) All string attributes are converted to nominal attributes to work around a limitation in our datamining toolkit.
4) All file system metadata timestamps (`crtime`, `ctime`, `mtime`, and `atime`) are removed from the test set, as these timestamps are not available in carved data.[1]

E. Building and Testing the Classification System

Once metadata is extracted from all the exemplars, we train a set of machine learning classifiers using Weka [20], a freely available program that implements customizable versions of a wide variety of machine learning algorithms.[2]

After preprocessing, we train and test multiple classifiers on the data using leave-one-out validation. (Please see Section III-G to explain our choice of leave-one-out validation.) Because metadata is not directly associated with the process that overwrites directory entries (which forces undeleting/carving of files), we postulate that carved files, undeleted files, and allocated files will all have a similar distribution of metadata. We derive from this postulate the implication that validation accuracy is the best measure of the success of the classifier.

We display the results of the validation runs in a confusion matrix in Tables VII –XII. Each row of these tables shows how the files that were ascribed to each user would be ascribed if the ownership information is removed: the boxed number along the diagonals of the tables present the correct classifications. The last row of the table presents the percentage of correct classifications for each classification group. We also compute the "average accuracy" of the classifier, which we define as the average of the individual classification accuracies for each user.

In actual use to ascribe carved data we would finally construct a classifier using *all* of the exemplars in the disk image. `fiwalk` would be run again, this time with a special "carving" input file that documents the specific sector locations inside the disk image containing information to be carved. (The format of this carving input file is identical to the output format of the log file created by the popular Scalpel file carver [10].)

F. Classifiers Used: KNN and C4.5

We tested two classification algorithms, K-Nearest Neighbor (KNN) and C4.5 decision trees (called "J48" by Weka). These two algorithms were selected for their ability to deal with high-dimensionality, heterogeneous data sets in which instances of a class are not contiguous, which is characteristic of our input data. For instance, one user may use the computer for an hour, followed by another user for two hours, and then a return to the first user, resulting in two distinct noncontiguous timelines when a file is likely to be created by the first user.

1) KNN: KNN is a simple data mining technique in which all data elements are mapped into a multidimensional space. Input elements are classified according to the type of the nearest

[1] File system metdata is available for orphan files; experiments not presented here indicate that using this additional metadata increases accuracy.

[2] To facilitate interoperability with Weka, we modified `fiwalk` so that it could directly output the results of the extraction as an Attribute Relation File Format (ARFF) file.

exemplar (in the case of $K = 1$), or some kind of voting function of the N nearest-neighbors. (See Dasarathy [3] for a complete discussion.) We consider KNN's simplicity to be an advantage when attempting to explain this process to a magistrate or a jury. It is easy to explain that a JPEG file in question is likely to belong to the defendant because it has the same serial number or camera settings as other files that are known to belong to the defendant, or that a Word file is likely to belong to a person because it has a document print-time that is within 3 seconds of another file's print time. We believe that it would be harder to explain a complex decision tree. At the same time, it can be seen as additional confirmation when both KNN and C4.5 produce the same conclusion.

The version of KNN that ships with Weka uses a class known as `EuclideanDistance` for determining the distance between two instances. This distance function has two deficiencies that made it perform poorly here. First, missing values result in a maximal distance for the dimension of the missing value. Second, timestamps, which are the *a priori* most promising feature for file ascription, are ignored by Weka's `EuclideanDistance` distance function. With these issues in mind, we crafted a replacement class for measuring distance, called `ReasonableEuclideanDistance`. Our distance function ignores missing values in distance calculations, and converts time stamps into seconds. All dimensions are then normalized to the range [0,1].

2) C4.5 Decision Trees (J48): J48 is an open-source implementation of Quinlan's C4.5 algorithm for developing decision trees [17]. The algorithm attempts to find attributes in the data that split the exemplars into subsets that have more variation in the parameter to be classified than the original pool. Although C4.5 produced better results than KNN, it has two distinct disadvantages:

a) It is necessary to build each C4.5 classifier based on the specific attributes available in the carved data: it makes no sense to build a highly accurate classifier that uses the Microsoft Office "Last-Printed" time if one is attempting to classify a JPEG for which Last-Printed time is not available.

b) Although one of the attractive features of a decision tree is the ability to explain the reasoning behind its predictions, in our runs, the algorithm produced decision trees that were perplexing and could be difficult for even an expert to explain (Fig. 2). This might complicate the use of the C4.5 algorithm in a court room.

Because KNN may be legally superior while C4.5 produces superior validity, we present the results of both algorithms throughout this paper. For this reason, we recommend presenting the results of both algorithms.

A reviewer of a previous draft of this paper suggested that perplexing decision trees may be a result of over-fitting. Although this may be the case, we believe that over-fitting is not a relevant concern for the classification task that we present in this paper.

Overfitting is a risk in machine learning when training data is rare or when there are random processes that are present in the training data that is not present in the test data that is being classified. However, in this case, the training data and the test data are drawn from the same sample—the subject's hard drive. In many cases, the only difference between the training data and the test data is that the test data was in a file that was deleted and the file's directory entry was substantially overwritten. The test data was subject to the same seemingly random processes as the rest of the training data. Thus, overfitting is not a concern.

G. Classifier Validation

Data mining performance is typically evaluated using ten-fold cross validation [11], a process in which the data is divided into ten equal parts (or "folds"). Ten classifiers are then created, each with one tenth of the data held out. Each classifier is then used to classify the data that is held out, the process being repeated ten times.

Leave-one-out is an alternative validation approach. This approach works similarly to ten-fold cross validation, but instead of holding out 10% of the data as a test set, a single instance is held out each time, and each other instance is used to train the classifier. This approach is less favored than the ten-fold approach because it takes significantly longer to perform, and because it produces a result that is highly biased to the specific data set being used by the classifer.

We feel that leave-one-out validation is a more accurate model to our problem at hand, since we are not attempting to create a general purpose classifer: we are trying to create a general purpose classification *technique*, but the classifier produced by this technique will only be used *once*—to ascribe data based on the very hard drive from which the classifier was created.

Given the high proportion of training data to test data, leave-one-out (in this scenario) produced significantly better classification rates than ten-fold cross validation. For instance, if a user only used the computer to create a handful of files during a certain time span, and many or all of those files are omitted in the 90% used to train the classifier, then the classifier will perform poorly on the test set. This can result in suboptimal results from the the classifier.

Ten-fold validation is commonly used to evaluate data mining, largely because the datasets are known, and consistent comparisons are necessary to fairly compare efficacy of new algorithms and approaches. Leave-one-out is a better match to the problem at hand, which must stress accuracy over arbitrary performance concerns. However, for the purpose of comparison and establishing baselines, we present side-by-side results in Table III.

IV. Results

Our initial experiments were conducted using the realistic (and freely redistributable) "domexusers" and "seed1" drive images from the http://digitalcorpora.org [7] project (Table II). The "domexusers" disk was created by a researcher alternatively playing the role of two different computer users that were using a computer to communicate with a third person. Each fictional user was given an account; the experimenter toggled between the two accounts using the "fast user switching" feature built in to Microsoft XP. The "seed1" disk was created by multiple

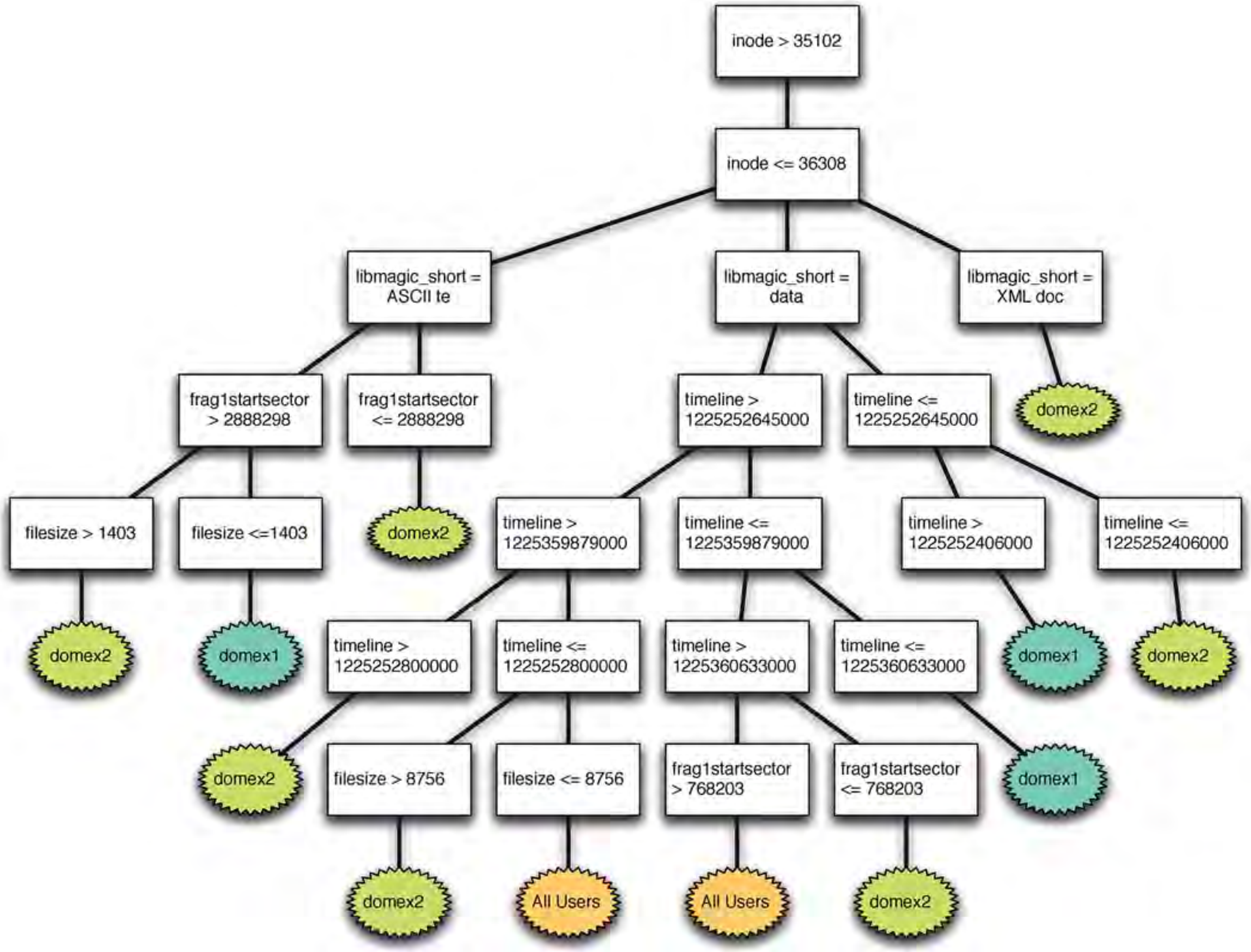


Fig. 2. C4.5 decision tree fragments without and with timeline data.

individuals who sat down at a computer over the course of several weeks, each one alternatively browsing the web and downloading documents. Table III presents the results of the basic approach applied to these two disk images.

Working with realistic but synthetic data has two significant advantages:

1) Since the data can be freely downloaded from the Internet, synthetic data allows others to replicate our results and attempt to improve on our algorithms [7]. (Although our real data sets are available to other researchers, the use of that data set requires that the researchers first receive permission from their organization's Institutional Review Board.)
2) The use of synthetic data allows us to report the results here without the need to obscure or redact any information (as was necessary for our result tables based upon real data). For example, some organizations have custom metadata fields their Microsoft Office documents that are injected by document control systems or custom-written applications. If the metadata fields present in Table I were based on real data, rather than synthetic data created in a laboratory, the presence of those fields might disclose the name of an organization from which the drive had been obtained. This could have negative impacts on both the organization and one or more of its employees.

The KNN classifier performed poorly with Weka's Euclidean distance metric, but well using our "Reasonable" distance metric discussed above (see Table III). Accuracy was best with $N = 1$ and decreased as N increased, evidence that instances of a specific class are rare in any given neighborhood. Our use of KNN was "naïve," in the sense that we chose to use each possible metadata attribute as a separate (and equally important, given the Euclidean basis of proximity measurement) dimension. The poor performance of ten-fold cross validation is further evidence of the sparseness of points within the multidimensional space, a shortcoming of KNN for this specific problem.

The C4.5 classifier performed quite well in our initial runs. In our examination of the resulting classifier trees, we were surprised that features such as starting disk sector number were used in preference to features such as the Microsoft Office embedded "Creator" attribute. We believe that this is because the C4.5 algorithm builds its classifier by organizing decision nodes in order of decreasing entropy from the root. On a device with a wide variety of files, only a few will be of each type, and any em-

TABLE II
DRIVES USED IN EXPERIMENTS, ALONG WITH THEIR OPERATING SYSTEMS AND FILE SYSTEM TYPE. FILE SYSTEM USAGE FOR USERS WITH THE MOST NUMBER OF FILES IS ALSO GIVEN (USER #1 IS THE USER WITH THE MOST FILES); FULL INFORMATION REGARDING FILE SYSTEM ALLOCATION CAN BE FOUND IN THE CONFUSION MATRIX AT THE END OF THIS PAPER

		File	#	#	# files per user (% of total files)		
Drive ID	Host OS	System	attributes	users	user #1	user #2	user #3
Drives with realistic laboratory-generated data:							
domexusers	Windows XP	NTFS	163	7	1304 (48%)	797 (29%)	617 (23%)
seed1	Windows XP	NTFS	183	9	1286 (29%)	1140 (25%)	932 (21%)
Drives with real data, originally purchased on the secondary market:							
0844	Windows XP	NTFS	132	14	20856 (49%)	20692 (48%)	364 (1%)
mx4-18	Windows 2000 SP4	NTFS	148	9	6536 (38%)	5034 (29%)	4937 (29%)
mx5-24	Windows NT 4 SP5	NTFS	68	8	797 (61%)	206 (16%)	103 (8%)
mx7-03	Windows 2000 SP3	NTFS	108	10	1338 (35%)	1256 (33%)	682 (18%)

bedded metadata field will be rarely populated—for example, although "Creator" will be present for Microsoft Office files, it is not present for JPEG files. This leads to low entropy for embedded attributes over all records, and produces a classifer that ignores the very attributes that a human analyst would consider to be the most predictive. Sector number, on the other hand, is loosely correlated with creation time as the drive is filled, and is thus correlated with the computer's operator. A revised C4.5 algorithm specifically designed to work with sparse data sets with many missing values might score even higher.

We hypothesize that better accuracy could be obtained by making use of file system timestamps. After all, there is a wealth of information on these disk images that is being simply thrown away. Our initial implementation ignores `crtime`, `ctime`, `mtime`, and `atime` because these attributes are not available for carved data. We discuss how to make use of these timestamps in Section V.

V. USER ACTIVITY TIMELINE

There are many attributes that are timestamps which result from the activity of a user sitting in front of a computer system. For example, double-clicking on a file causes the file's `atime` to be updated when it is read. Making a change to the file and saving it causes the file's `mtime` to be updated. Many file types further have embedded timestamps. Microsoft Office files contain an embedded timestamp of when the file was last saved; JPEGs contain a timestamp of when the image was "digitized," which reflects either when the photograph was snapped or when the file was edited and saved using an image manipulation tool.

In Section IV, we treated each of these timestamp types as different dimensions. Yet many of the timestamps correspond to activity caused by a user sitting in front of a computer system and manipulating the mouse and keyboard.

In this section, we explore the performance improvement that results by projecting all of these times onto a single dimension to create a *user activity timeline*.

A. Basic Approach

Many file types contain multiple timestamps. The most straightforward way to use these timestamps is to project them onto a single timeline indicative of user activity. To do this, we create a "unified" timeline that contains all of the timestamps found in ascribable exemplars. If an HTML file belonging to User A was found to be created at 7:05 P.M., modified at 7:06 P.M., and last accessed at 7:10 P.M., three points associated to User A will be added to the timeline. Microsoft Office files may result in up to eight points being added: four points for the NTFS file system timestamps, and four points corresponding to embedded timestamps.

The power of the unified timeline is that carved data frequently contains embedded timestamps. By creating a single unified timeline, we can use file system timestamps to help ascribe ownership of carved data, even though file system timestamps are not available for carved data.

Consider the case of a carved Microsoft Word file that is found to have been printed at 10:33 A.M.. Without the unified timeline, only the print times of other Microsoft Word files could be used for ascription: if no other Microsoft Word files on the hard drive had been printed around 10:33 A.M., this information would be ignored. With the unified timeline, a Notepad file that was created at 10:32 A.M. could be used to ascribe ownership to the carved Microsoft Word file, since the print-time of the Word file and the create time of the Notepad file would be projected onto the single timeline.

Note that the creation of a unified timeline multiplies the number of items to be classified, as each file can have a maximum of eight times, each of which projects to a different point on the timeline. As a result, the confusion matrices presented at the end of this paper have more items that are classified than the number of files on each device.

Note also that the critical assumption of our user activity timeline is that only one person is accessing the system at a time. We have not characterized the behavior of the algorithm when there are simultaneous accesses to the file system by different users.

B. Timestamp Suppression

Care must be taken when reducing timestamps to a single dimension, as all timestamps cannot be treated equally:

- Invalid timestamps must be detected and suppressed. Our system discards timestamps with a date of 0000-00-00 (zero) or 1970-01-01 (the Unix epoch).
- Timestamps originating on systems other than the target system need to be ignored. For example, timestamps embedded within HTML files should be suppressed, since they originated on a remote web server. Likewise, timestamps embedded within JPEGs should be suppressed if they originated within a digital camera, as activity within a

TABLE III
CLASSIFICATION AVERAGE ACCURACY FOR NEAREST NEIGHBOR AND C4.5 ALGORITHMS, AS REPORTED BY LEAVE-ONE-OUT VALIDATION AND TEN-FOLD CROSS-VALIDATION. REPORTED ACCURACY IS THE AVERAGE OF CLASSIFICATION ACCURACY FOR EACH USER. ALL STATISTICS ARE COMPUTED USING "REALISTIC" (LABORATORY-CREATED) DISK IMAGES

Drive ID	Validation	1-NN Euclidean	1-NN Reasonable	3-NN Reasonable	10-NN Reasonable	C4.5
domexusers	leave-one-out	17.32%	78.45%	75.35%	68.44%	89.85%
domexusers	10-fold	13.37%	28.48%	27.46%	24.36%	39.90%
seed1	leave-one-out	22.62%	76.38%	69.03%	65.15%	85.60%
seed1	10-fold	9.72%	25.25%	25.96%	24.29%	25.24%

digital camera is unlikely to be correlated with activity on the target computer system.

As a result, our timeline system incorporates a number of media-specific rules, detailed below. Once the timestamps have been collected and sanitized, a single timeline of system usage can be created, and this new dimension of aggregated timestamps can be used to create a new classifier.

1) Timestamps Within Microsoft Office Rule: Microsoft Office stores timestamps in four specific metadata fields: Created, Editing-Duration,[3] Last-Print and Last-Modified. These are all activities that can be used as part of the user activity timeline.[4] Because these timestamps are created and maintained by the office application, rather than the file system, the Office document creation and modify times can be significantly different from the file's creation and modify time. Projecting both sets of timestamps onto the user activity timeline makes sense.

2) Timestamps in JPEG EXIF Structures Rule: JPEGs can contain several timestamps within the EXIF metadata segments, including "Date and Time," "Date and Time (original)," "Date and Time (digitized)," and possibly others.

Because the goal is to create a timeline of computer usage, timestamps originating within digital cameras should not be projected onto the common timeline of computer usage.[5] In order to filter out these camera-generated timestamps, our system examines the EXIF "Manufacturer" and "Model" tags and only keeps JPEG timestamps if the tags contain the string "Adobe," "Windows," "Macintosh," or "QuickTime." In our testing, the presence of these strings indicates that the JPEG was processed by image editing software, allowing us to assure that only the timestamps that result from editing on the host computer are projected onto the unified timeline.

3) Timestamps From Web Browsers Rule: When a file is downloaded using a web browser, ideally, we would like the timestamp to be set to the time of download. But not all browsers set the time in this fashion. Sometimes the modification time is set to the time that it was downloaded or written into the browser cache, which would be most useful for the ascription task. Sometimes the modification time is set to be the time provided by the webserver for the document when the document is downloaded by HTTP. Table IV documents how the behavior of different browsers impacts the timestamps of downloaded files.

TABLE IV
ACTIONS WHICH PRESERVE TIMESTAMPS RATHER THAN UPDATING THEM

Activity	OS	Preserved
Safari file download	MacOS	`mtime` of downloaded file reflects HTTP timestamp of URL from webserver
Safari file download	Windows	all timestamps reflect HTTP timestamp of URL from webserver
Finder copy	OSX	retains modify time, Birth time, and last opened time of the source file
File Manager copy	Windows	preserves `mtime` from source file
command line copy	Windows	preserves `mtime` from source file
GNOME copy	Linux	preserves `mtime` from source file

Although it is frequently possible to determine which web browser downloaded a particular file (for example, by examining the browser-specific cache or log files), at this time our system ignores this information. As a result, downloaded files can introduce inaccuracies into the timeline. These inaccuracies could be corrected with additional work.

A deployed system could suppress timestamps on files that had been downloaded using Safari but retain them on files downloaded by Firefox or Internet Explorer. The decision whether or not to suppress a timestamp can be made based on the filename, the containing directory, and the installed software.

4) Timestamps From File Copying Rule: Even the simple an act of file copying can introduce discrepancies into the timeline. NTFS, for example, tracks the "Create," "Modify," "Access," and "Entry Modified" time for each entry in the file system.[6] With most modern systems, copying a file through the graphical user interface preserves the file's modification time. Unfortunately, preserving the modification time is not wanted here:

[3] Confusingly, even though the name "Editing-Duration" would seem to imply a value of time duration, the actual value stored in Word documents is an absolute time.

[4] While it is likely that other office suites also store timestamps in their metadata, we did not examine these file formats, as the usage of suites such as OpenOffice and KOffice is dwarfed by Microsoft Office.

[5] Of course, if a single camera is used by multiple individuals, the techniques presented in this paper could be applied to a single camera.

[6] "Entry Modified" is the time that the file's entry in the Master File Table was modified. It is roughly equivalent to inode `ctime` in the Unix file system.

TABLE V
AVERAGE ACCURACY OF THE CLASSIFIERS ON "REALISTIC" DISK IMAGES WITH A UNIFIED TIMELINE (SEE SECTION V)

		1-NN	1-NN	3-NN	10-NN	
Drive ID	Validation	Euclidean	Reasonable	Reasonable	Reasonable	C4.5
domexusers	leave-one-out	22.72%	77.08%	77.09%	75.86%	91.28%
seed1	leave-one-out	20.82%	77.49%	76.40%	68.65%	92.15%

TABLE VI
APPLICATION OF THE UNIFIED TIMELINE METHOD TO REAL DISK DRIVES PURCHASED ON THE SECONDARY MARKET. ACCURACY IS COMPUTED WITH LEAVE-ONE-OUT VALIDATION. "AVERAGE ACCURACY" IS THE AVERAGE OF THE ACCURACY OF CLASSIFICATION FOR EACH USER. COMPLEX CONFUSION MATRICES FOR THESE RUNS CAN BE FOUND AT THE END OF THIS PAPER

	Average Accuracy	
Drive ID	1-NN	C4.5
0844	71.90%	91.81%
mx4-18	59.02%	88.37%
mx5-24	75.43%	96.47%
mx7-03	75.22%	91.14%

if User 2 copies a file from User 1's directory to User 2's directory, we would like that file to have a modification time reflecting when the file was moved by User 2, not when the file was modified by User 1. Most operating systems do not provide this functionality, and thus the `mtime` of files that are copied (or moved) from one user's directory to another user's directory can introduce inaccuracies into the ascription project. We have not determined an appropriate rule for handling this case, although one may not be needed as file permissions normally prohibit such copying. Table IV documents how different operating systems manage the modification or preservation of timestamps.

C. Results With the Unified Timeline

We reran the experiments using a single unified timeline in addition to the original features using the techniques discussed above. We found that reducing the dimensionality and merging semantically similar time information makes the classifier more accurate in all cases (Table V).

We believe that the reason for this improvement is that there is a high degree of clustering on the time axis. As a result, adjacent points on the timeline have good predictive value for the owner of an unknown point.

In examining the decision trees produced by the C4.5 algorithm, we noted that time was clearly the most important feature in determining which user to ascribe ownership. In several cases, other features were not even used. Intuitively this makes sense, especially on a desktop computer, where there is usually only a single user in control at any given time. Thus, timestamps are highly correlated with specific users.[7]

VI. RESULTS WITH REAL DATA

After improving the results with the user activity timeline, we tested the technique using disk images from the Real Data Corpus [7], a collection of more than 2000 disk images made from hard drives that were purchased on the secondary market. We selected images with intact file systems for which multiple users could be readily identified. Four appropriate multiuser disk images were identified (Table II). All were taken from computers running Microsoft Windows and were formatted with NTFS. Although NTFS has additional file system metadata including access control lists, security identifiers, and a sequence number, we did not use this information for our experiments, so the use of NTFS does not impact our results.

(The overwhelming number of disk images in the RDC are from camera cards, CDs, DVDs, and single-user computers. Of the multiuser computers that we found, many computers could not be used in this research project because the drive data originated within the United States.)

The results are summarized in Table VI and presented in Tables VII –XII. In general, we found that some drives produced classifiers with higher accuracy rates than others, but that most of the drives produced classification rates that would be high enough to be useful in an investigation.

VII. DISCUSSION AND FUTURE WORK

The confusion tables that we present confirm the hypothesis presented in Section I-A and show that the method presented here can achieve high accuracy rates under a variety of realistic and real world conditions.

A. Implications for Law Enforcement

Our complete analysis of the "domexusers" drive (Table VII) shows that the files on the drive can be ascribed to six different users: in addition to "domex1" and "domex2," there are also files belonging to the Administrator user, to the windows "All users" user, the "Default User" user, and to two system services, "LocalService" and "NetworkService." The bottom row of this table shows that classifications into these categories were accurate more than 80% of the time in all cases, and more than 95% of the time for files classified as "Administrator," "domex1" or "domex2." Similarly high rates of classification were seen for the seed1 (Table VIII), 044 (Table IX), mx7-03 (Table X), and the mx5-24 (Table XI) drives.

The mx4-18 (Table XII) drive, on the other hand, showed a relatively low rate of classification accuracy (66.53%) for files classified as "All Users." Even files classified as "g" were correct only 89.78% of the time.

These results have several important implications for the use of this technique in law enforcement or other analogous situations:

1) **Different hard drives will have different rates of classification accuracy**. The classification accuracy of this technique depends not just on the technique itself, but on the actual drive being classified. Some drives will classify well, while others will not. One of the strengths of this technique is that we can determine the overall classification accuracy for the drive.

[7] Although it is possible to set up remote logins on a desktop machine and consequently have multiple concurrent users, in practice both Windows and MacOS prohibit multiple simultaneous use of the graphical user interface due to licensing and security concerns.

TABLE VII
DOMEXUSERS (C4.5) CONFUSION MATRIX

	Classified As							
User	a	b	c	d	e	f	g	total
a “Administrator”	5118	62	0	26	4	7	4	5221
b “All Users”	57	1422	17	32	12	4	0	1544
c “Default User”	1	39	392	0	0	0	4	436
d “domex1”	21	62	0	3051	96	0	0	3230
e “domex2”	24	16	0	94	2335	0	0	2469
f “LocalService”	12	0	0	0	0	64	0	76
g “NetworkService”	2	2	0	0	0	4	48	56
% correct classifications	97.77	88.71	95.84	95.25	95.42	81.01	85.71	

TABLE VIII
SEED1 (C4.5) CONFUSION MATRIX

	Classified As									
User	a	b	c	d	e	f	g	h	i	total
a “Administrator”	3855	82	18	30	0	1	1	0	8	3995
b “All Users”	82	2061	23	64	29	64	7	4	22	2356
c “Default User”	9	16	1999	1	0	3	0	0	0	2028
d “dph2007”	21	64	2	3187	17	10	0	0	7	3308
e “hunter”	0	29	0	15	2110	15	0	3	0	2172
f “jjm2007”	4	21	0	11	77	3261	2	0	0	3376
g “LocalService”	0	14	0	0	0	2	179	1	0	196
h “NetworkService”	0	1	2	8	1	4	7	45	0	68
i “simsong”	16	8	0	7	0	2	0	1	4472	4506
% correct classifications	96.69	89.76	97.80	95.91	94.45	97.00	91.33	83.33	99.18	

TABLE IX
0844 (C4.5) CONFUSION MATRIX; NONSYSTEM NAMES ARE BLINDED

	Classified As														
User	a	b	c	d	e	f	g	h	i	j	k	l	m	n	total
a “Administrator”	320	8	0	1	11	0	3	1	0	0	0	0	0	0	344
b “All Users”	13	971	0	3	11	0	15	7	0	18	3	1	24	2	1068
c (Blinded)	0	0	443	1	1	4	20	0	0	15	4	0	4	0	492
d (Blinded)	0	8	1	288	0	0	82	0	0	0	0	0	1	0	380
e “Default User”	8	36	4	0	343	1	4	0	4	0	4	0	0	0	404
f (Blinded)	0	0	12	0	1	440	4	0	3	0	0	0	0	0	460
g (Blinded)	2	8	22	46	6	11	66540	2	8	23	13	0	7	2	66690
h “LocalService”	5	2	0	0	0	0	4	75	0	0	0	0	2	0	88
i (Blinded)	0	0	1	0	1	0	0	0	707	0	2	0	1	0	712
j (Blinded)	0	12	4	2	0	1	22	1	0	594	0	0	0	0	636
k (Blinded)	0	3	4	0	1	0	8	0	0	0	1204	0	0	0	1220
l “NetworkService”	0	0	0	0	0	0	0	2	0	0	0	54	0	0	56
m (Blinded)	0	16	0	0	0	0	8	6	2	0	0	0	70952	224	71208
n (Blinded)	0	8	0	0	0	1	0	0	2	0	0	0	81	436	528
% correct classifications	91.95	90.58	90.22	84.46	91.47	96.07	99.75	79.79	97.38	91.38	97.89	98.18	99.83	65.66	

2) **Different users on a single hard drive will classify at different rates**. Some users may a high regularity to the information that they deposit on the drive, while others may have little or no regularity. Again, one of the advantages of this technique is that we can infer, from the bottom row of the confusion matrix, the accuracy of each kind of classification determination. For example, on the drive mx4-18, only 91.47% of the files classified as belonging to “administrador” actually belonged to this user, which means that files ascribed to “administrador” have roughly a 1-out-of-10 chance of being incorrectly ascribed. Were this technique presented at a trial, this error rate should rightfully be presented in court to allow the jury, judge, or ultimate trier of fact to make an informed decision.

We believe that the correct way to address these issues is to present the per-classification accuracy rates when using classification results in a legal context.

TABLE X
MX7-03 (C4.5) CONFUSION MATRIX; NONSYSTEM NAMES ARE BLINDED

	Classified As										
User	a	b	c	d	e	f	g	h	i	j	total
a (Blinded)	335	0	0	0	4	0	33	0	0	0	372
b “Administrator”	0	2224	72	20	0	34	33	8	0	8	2399
c “All Users”	2	72	1354	16	0	35	99	6	0	16	1600
d “Default User”	0	19	12	225	0	0	0	0	0	0	256
e (Blinded)	1	1	0	0	355	0	3	0	0	0	360
f (Blinded)	0	46	56	0	0	4332	16	80	0	72	4602
g (Blinded)	44	27	87	0	4	16	11096	11	5	8	11298
h “nocuser03”	0	1	15	0	0	51	2	356	0	7	432
i “nocuser04”	0	1	0	0	0	0	8	3	332	0	344
j (Blinded)	0	0	14	0	0	65	10	6	0	597	692
% correct classifications	87.70	93.02	84.10	86.21	97.80	95.57	98.19	75.74	98.52	84.32	

TABLE XI
MX5-24 (C4.5) CONFUSION MATRIX; NONSYSTEM NAMES ARE BLINDED

	Classified As								
User	a	b	c	d	e	f	g	h	total
a “Administrador”	748	5	0	19	1	0	0	0	773
b “All Users”	12	967	0	9	30	2	0	0	1020
c “CORPORATIVO”	0	0	277	0	0	0	0	0	277
d “Default User”	5	20	0	164	0	0	0	0	189
e (Blinded)	5	20	0	0	2591	4	0	0	2620
f (Blinded)	0	0	0	0	1	412	4	0	417
g (Blinded)	0	0	0	0	0	4	273	4	281
h (Blinded)	0	0	4	0	0	0	0	277	281
% correct classifications	97.14	95.55	98.58	85.42	98.78	97.63	98.56	98.58	

B. How Much Accuracy Is Needed?

An anonymous reviewer of a previous version of this paper asked “Is 90%–95% accuracy sufficient for courtroom evidence?” This is an excellent question. Unfortunately, it is a question that cannot be answered.

Although the U.S. Supreme Court held in its *Daubert* ruling that scientific evidence can only be presented if there is a “known error rate,” [18], most digital forensic techniques do not have an error rate [13]. These tools lack an error rate because they either work or they do not work, so their perceived error rates are either 0% or 100%.

One of the significant contributions of the approach presented here is that *our tool reports an error rate!* Significantly, it reports an error rate that is specific for the disk on which it is run. This is not a deficiency of the technique. On some multiuser disks, there will be significant differences between the activity patterns of each user, while on other disks, there will be no discernible difference. The fact that our technique can report when it is likely to be reliable and when it is not reliable is a significant advance for digital forensics.

C. Attacks on this Technique

As with other computer forensic techniques, there are many ways that a knowledgeable adversary could frustrate this technique or subvert it to frame an innocent person. The most straightforward way to frustrate this technique would be to use disk wiping or full disk encryption to prevent the extraction of file placement information and metadata from the storage device. Programs such as CCleaner [12] and Apple’s Disk Utility can also be used to erase residual information in unallocated space.

The system presented here relies on the operating system to distinguish which exemplars were actually used by each system user. This approach will not work if one user manages to obtain control of another user’s account, either by learning their password or through the use of malicious software. Likewise, it will not work for the case where multiple users share the same account, rather than merely sharing the same user (see Section VII-D1). Of course, these risks are already present for any forensic examination: the work presented here merely extends this risk to the automated ascription of carved data.

Finally, an attacker could create patterns to mis-train the classifier, resulting in attribution of a file or its remnants to the incorrect owner. Fundamentally, this is no different than an attacker planting digital evidence that implies a different user is guilty of the crime. This is a problem for digital forensics in general and is not specific to the technique presented here.

As with other forensic techniques, the existence of these attacks does not make the contribution of this technique less useful. Few perpetrators make use of anti-forensic tools today, and even fewer plant digital evidence with the hope of misdirecting investigations. While this technique makes it possible for a perpetrator to frame an innocent party, so does

TABLE XII
MX4-18 (C4.5) CONFUSION MATRIX; NONSYSTEM NAMES ARE BLINDED

	Classified As									
User	a	b	c	d	e	f	g	h	i	total
a "administrador"	590	5	12	0	0	0	12	10	0	629
b "All Users"	10	1282	211	12	0	0	376	35	2	1928
c (Blinded)	12	204	27216	1	3	5	750	36	0	28227
d "Default User"	0	11	2	174	2	0	3	1	0	193
e (Blinded)	0	1	2	3	951	4	48	0	0	1009
f (Blinded)	0	0	1	0	7	223	10	0	0	241
g (Blinded)	6	409	5521	3	171	31	13970	125	1	20237
h (Blinded)	27	13	167	0	4	0	391	16395	6	17003
i (Blinded)	0	2	1	0	0	0	1	14	455	473
% correct classifications	91.47	66.53	82.14	90.16	83.57	84.79	89.78	98.67	98.06	

every other digital forensics technique in use today. This technique is no more susceptible to framing and fraud than other well-established techniques. In fact, this technique may be less susceptible, because it uses such a large amount of data on which to base its conclusion, the steps in the deductive process are made visible, and because the system computes an error rate for each use.

D. Future Work

In addition to its benefits for ascribing deleted files for law enforcement purposes, there are some other applications of this work. For instance, identifying orphaned/explicitly deleted and overwritten files that left behind inode information is useful. The techniques can also be used on files in the file system (i.e., not deleted) to ascribe files on drives that are outside the user's home directory.

1) Two Users Sharing One Account: One key problem with file ascription is the use of a single login by multiple users. The technique presented in this paper could be modified to determine if two or more users are sharing a single account by detecting differences in file patterns associated with different times of day.

In a related scenario, suppose a computer is stolen, used by the thief, and then recovered. This technique can be modified so that the exemplars are segregated not just by account name, but by time. This would be a straightforward modification of our work to date.

2) Application to Noncarved Data: Although this paper is primarily concerned with the ascription of ownership to data that has been carved, the techniques presented in this paper can be applied equally well to allocated data that is located on a multiuser system that cannot be readily ascribed to one user. This might be the case, for example, if the data is stored in a directory that does not belong to a specific user and for which no file ownership information is available.

3) Timeline and Geographical Correlation: Our initial set of rules for processing timestamps could be extended to track multiple timelines in multiple locations—for example, if the dataset contains data from three different computer systems, it would be appropriate to create three separate timelines. Likewise, it might be fruitful to mine the system for locale or geospatial information and use these points for correlation.

4) Ascribing Portable Storage Devices and Cloud Storage: There is no reason that the techniques we put forth here need to be limited to a single hard drive. With the exception of location on a specific hard drive (e.g., the sector number of the first fragment), all of the metadata would be available for files stored on portable storage devices such as USB storage devices and SD cards. Extending this technique to handle multiple media should be a simple matter, although it will be necessary to either detect and correct for time shifts caused by multiple clocks, or else to avoid using time for ascription purposes. This technique could likewise be used to ascribe information stored in the Internet "cloud." In this case, the automation would be used to extract features and automatically compare the stored document with exemplars in a reference collection.

5) Validation Server: Although the technique presented in this paper is effective, it is time consuming to train and validate a classifier before using it to ascribe a file. For example, drive 0844 contained more than 20 000 files from each of the two primary users. Our experimental system required less than 30 s of CPU time to perform each leave-one-out validation, but this translated to approximately a week of CPU time on a single-processor machine.

For experimental purposes, we used a large-scale (100+ node) grid computer, but such systems are not necessary. With the decrease in cost of multiprocessor machines with multicore processors, the performance problems of leave-one-out validation become less severe. Once the clustering of the drive metadata had been validated and error rates determined, individual ascription runs could be performed on a typical desktop computer.

6) Validation With Statistical Sampling: A typical multiuser storage device may have tens of thousands or even hundreds of thousands of files that belong to each user. Performing leave-one-out validation in such a situation can be computationally expensive, even with a cluster.

As an alternative, we plan to explore the possibility of performing validation using statistical sampling. That is, instead of performing leave-one-out validation using *all* of the exemplars available for each user, we hope to evaluate the feasibility of using a sampling of randomly selected exemplars.

7) Dimensionality Reduction: While this technique performs quite well under the algorithm described, it may be possible to eke out further performance improvements by applying further dimensionality reduction techniques, such as Principal Component Analysis (PCA). This approach will only be effective if there is a high degree of correlation between attributes, something which remains to be determined.

8) Improvements Through Content Analysis: The technique presented in this paper ignores the use of content to perform ascription. The use of content could obviously be added to improve ascription performance.

9) SVMS and Other Classifiers: Although our analysis is confined to the KNN and C4.5 algorithms, there is clearly an opportunity to use more modern algorithms such as support vector machines (SVMs) or “Boosting” algorithms for the classification. KNN might be improved by using a weighted distance function, so that close items would count more, or by giving different weights to different attributes. It would also be useful to evaluate the LazyDT algorithm [4].

10) Improved Handling of File Fragmentation: Currently our system only concerns itself with the first fragment of files that have multiple fragments. It may be useful to separately process each fragment.

11) Recursive Analysis of Container Files: Most of the file types in use today are in fact *container* files that can hold other files. For example, the Microsoft Word file format is a container format, and when a JPEG is pasted into a Microsoft Word document the JPEG is simply embedded in the Word file’s datastream.

Although some computer forensic tools will recursively process container files for keyword searches, `fiwalk` will not recursively process such files for automated metadata extraction. Adding such a feature to `fiwalk` would probably increase the accuracy of this approach, as it would provide the classification algorithms with additional information with which to classify.

12) Similarity of Carved Data Metric: An underlying assumption of this research is that the carved data to be ascribed has a similar distribution of metadata to some of the ascribable data on the hard drive. This similarity can be quantified and the accuracy of the classifer can be further refined as a function of the distance metric.

One approach to classify the similarity would be to consider the distance to the nearest neighbor in absolute terms. Another would be to compare the distance between several nearest neighbors with the known owner of the ascribable neighbors. Still another approach would be to consider the distribution of nominal attributions within the carved data and see if they matches attributes in the training data. Applying any of these techniques may provide additional insight into the results of the KNN classification process.

VIII. Conclusion

This paper has demonstrated that metadata embedded inside files can be used to predict the owner of files with a high degree of accuracy, even if the file system metadata is unavailable. Using a forensic file system crawler and metadata extractors we can extract a single timeline of file system usage, along with other supporting metadata. This can be used, along with machine learning techniques, to discover exemplar files which are similar to a file of interest, and use them to attribute the file in question to a specific user. Furthermore, this technique can easily be validated by a human expert, and explained to a jury, by examining the generated timeline and the suggested exemplar files.

A. Code Availability

The program `fiwalk` that was used to extract file names and metadata from disk images can be downloaded from http://www.afflib.org/. `fiwalk` is a freely available open source forensic framework that ingests a disk image, automatically processes all partitions that might be allocated in the disk image, examines all of the files and inodes in each one, and outputs information about each inode in a variety of formats, including ARFF and XML. A copy of this paper and the ascription engine can be found in the “tools/ascription” directory of the fiwalk source code release.

The Weka data mining toolkit used for this project can be downloaded from http://www.cs.waikato.ac.nz/ml/weka/.

Acknowledgment

The authors would like to thank N. Beebe, J. Lehr, B. Malin, R. Meijer, and R.-J. Mora for their useful insights with respect to the problems presented here and their review of this paper. B. Allen reviewed a previous version of this paper. The authors also wish to thank G. Dinolt and B. Rosenberg for their guidance and support of this project. Special thanks to J. Haferman, E. Adint, and the Naval Postgraduate School’s Information Technology and Communications Services Department for their tireless work operating Hamming, the NPS High Performance Computing Center’s Sun Microsystems Blade Supercomputer. J. Cowan-Sharp contributed to some of the research presented in this paper.

References

[1] R. A. Bosch and J. A. Smith, “Separating hyperplanes and the authorship of the disputed federalist papers,” *Amer. Math. Monthly*, vol. 105, pp. 601–608, 1998.

[2] B. Carrier, The Sleuth Kit and Autopsy: Forensics Tools for Linux and Other Unixes 2005 [Online]. Available: http://www.sleuthkit.org/, accessed 06 March 2009

[3] B. V. Dasarathy, *Nearest Neighbor: Pattern Classification Techniques (Nn Norms : Nn Pattern Classification Techniques)*. Los Alamitos, CA: IEEE Comput. Soc. Press, 1990.

[4] J. Friedman, Y. Y. Yun, and R. Kohavi, “Lazy decision trees,” in *Proc. 13th Nat. Conf. Artificial Intelligence*, Portland, OR, 1996.

[5] S. L. Garfinkel, “Forensic feature extraction and cross-drive analysis,” in *Proc. 6th Annual Digital Forensic Research Workshop (DFRWS)*, Lafayette, IN, Aug. 2006, Elsevier.

[6] S. L. Garfinkel, “Automating disk forensic processing with sleuthkit, XML and Python,” in *Proc. Fourth Int. IEEE Workshop on Systematic Approaches to Digital Forensic Engineering*, Oakland, CA, 2009.

[7] S. L. Garfinkel, P. Farrell, V. Roussev, and G. Dinolt, “Bringing science to digital forensics with standardized forensic corpora,” in *Proc. 9th Annual Digital Forensic Research Workshop (DFRWS)*, Quebec, Canada, Aug. 2009.

[8] A. Haeberlen, P. Kouznetsov, and P. Druschel, “Peerreview: Practical accountability for distributed systems,” in *Proc. Twenty-First ACM SIGOPS Symp. Operating Systems Principles (SOSP’07)*, New York, 2007, pp. 175–188, ACM.

[9] D. Holmes, “Authorship attribution,” *Computers and the Humanities*, vol. 28, pp. 87–106, 1994.

[10] G. G. R. III and V. Roussev, “Scalpel: A frugal, high performance file carver,” in *Proc. 2005 Digital Forensics Research Workshop (DFRWS)*, New Orleans, LA, Aug. 2005.

[11] R. Kohavi, *A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection*. San Mateo, CA: Morgan Kaufmann, 1995, pp. 1137–1143.

[12] Ccleaner 2009, P. Ltd.

[13] J. Lyle, “If error rate is such a simple concept, why don’t I have one for my forensic tool yet?,” in *Proc. 10th Annual DFRWS Conf.*, Portland, OR, 2010.

[14] J. Migletz, “Automated Metadata Extraction,” Master’s thesis, Naval Postgraduate School, Monterey, CA, 2008.

[15] N. Mikus, “An Analysis of Disc Carving Techniques,” Master’s thesis, Naval Postgraduate School, , Mar. 2005.

[16] F. Mosteller and D. L. Wallace, “Inference in an authorship problem,” *J. Amer. Statist. Assoc.*, vol. 58, pp. 275–309, 1963.

[17] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA: Morgan Kaufmann, 1993.

[18] Daubert v. Merrell Dow Pharmaceuticals 509 US 579, 1993.

[19] S. Y. Willassen, “Methods for Enhancement of Timestamp Evidence in Digital Investigations,” Ph.D. thesis, Norwegian Univ. Science and Technology, Norway, 2008.

[20] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 2005.